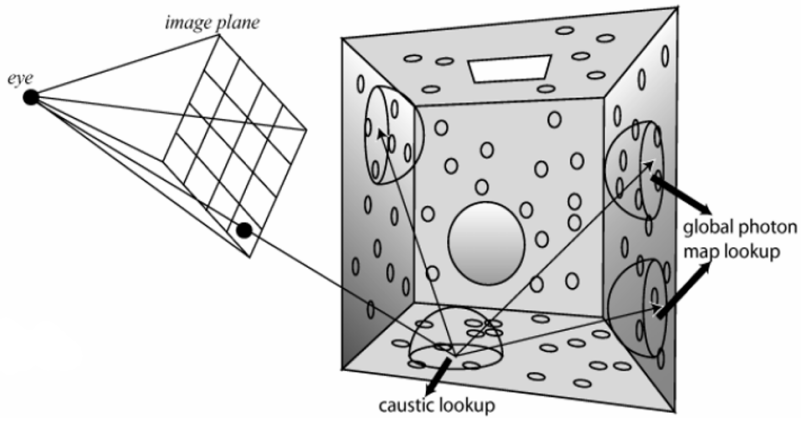
Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

INFORMÁTICA GRÁFICA

PhotonMapping



Alumnos:	Manel Jordá Puig Rubio	839304
	Ming Tao Ye	839757

11 de enero de 2025

Índice

1. Introducción **4**

2. Ecuación de Render **4**

3. Iluminación directa **5**

4. BSDFs Delta **7**

5. Análisis de los parámetros **9**
5.1. Número de fotones 9
5.2. Número de vecinos 10
5.3. Conclusiones 11

6. Diseño de la ruleta rusa **12**

7. Extensiones **13**
7.1. Kernels adicionales 13
7.2. Mapa de fotones cáusticos 15

8. Carga de trabajo y metodología **16**

9. Bibliografía **16**

1. Introducción

En este informe, se analizarán distintos aspectos del proyecto de *photon mapping* elaborado a lo largo del cuatrimestre para la asignatura de Informática Gráfica.

2. Ecuación de Render

Para explicar cómo se ha estimado la **ecuación de render**, en primer lugar mostramos la propia ecuación original junto con una representación conceptual de la misma.

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_\Omega L_i(x, \omega_i) f_r(x, \omega_i, \omega_o) |n \cdot \omega_i| \, d\omega_i$$

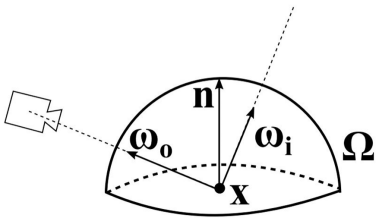


Figura 2.1: Representación visual de la ecuación de render. En la implementación, w_o está en el sentido opuesto.

Para resolver la integral de la ecuación se ha utilizado el mismo método de aproximación por **Monte Carlo** [1] ya mencionado en *path tracing*, con un muestreo de los rayos uniforme sobre el coseno (muestreo por importancia). Se ha utilizado también la misma aproximación que en el *path tracing*, tanto para materiales difusos como para materiales especulares y refractantes:

$$L_o(x, \omega_o) \approx \sum_{i=1}^{N} L_i(x, \omega_i) k$$

Esta aproximación ha sido utilizada en el primer paso del *photon mapper* para calcular el flujo de los fotones que rebotan por la escena, siendo L_i el flujo del fotón antes de rebotar (el flujo al salir de la luz es de $4\pi p_o/S$, siendo p_o la potencia de la luz (puntual) y S el número de fotones que han salido de esa luz). Como en ningún momento se ha llegado llenar hasta el límite sel mapa de fotones (no se pudiendo haber lanzado los S fotones totales) no ha hecho falta actualizar la S después de lanzar los fotones en ningún momento.

De nuevo, para decidir el evento asociado al rayo (difusión, reflexión, refracción o absorción), se utiliza el algoritmo de **Ruleta Rusa**[2].

En nuestra implementación, la luz incidente L_i directa en un punto sobre una superficie difusa puede calcularse utilizando la función **NextEventEstimation** (omitiendo además el primer fotón

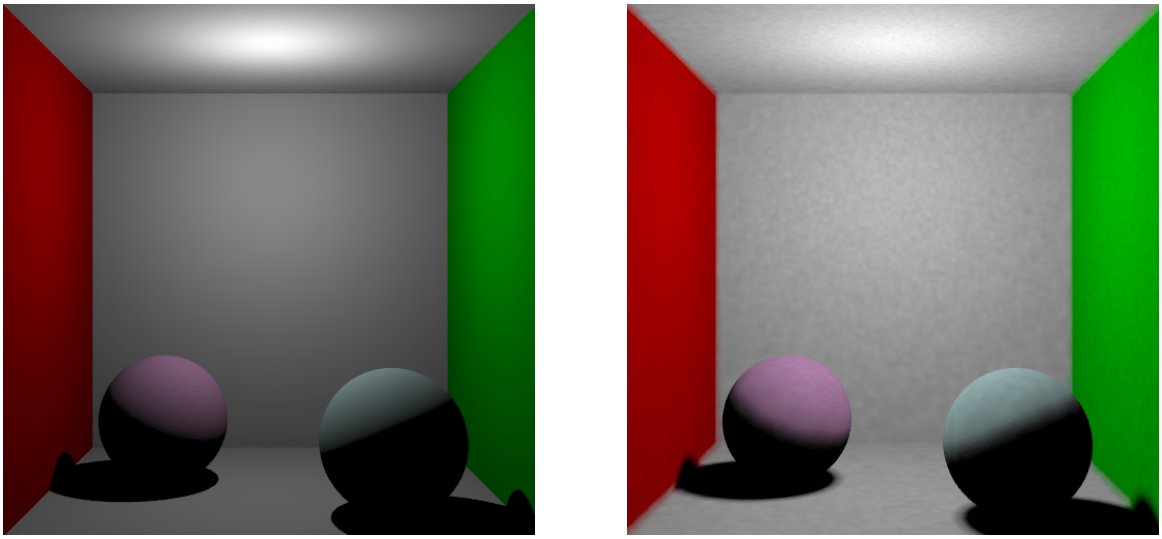
[1] Más información en: https://www.pbr-book.org/4ed/Monte_Carlo_Integration#

[2] En el apartado 6 se explican algunas decisiones tomadas para su diseño, que difieren con la ruleta rusa del *path tracing*

de cada *random walk* en el paso 1), siguiendo la misma aproximación mencionada en el *path tracing*. Sin embargo, para las pruebas, excepto en la figura 3.1a se ha optado por utilizar el primer foton almacenado en lugar de NEE.

3. Iluminación directa

A continación, se muestran dos escenas con iluminación directa. En la figura 3.1a se ha utilizado *Next Event Estimation* (sin fotones), y en la figura 3.1b se ha estimado la luz directa a través del primer fotón de cada *random walk*:



(a) Con NEE, 22 segundos (b) Con mapa de fotones, 5 minutos

Figura 3.1: Comparación de luz directa

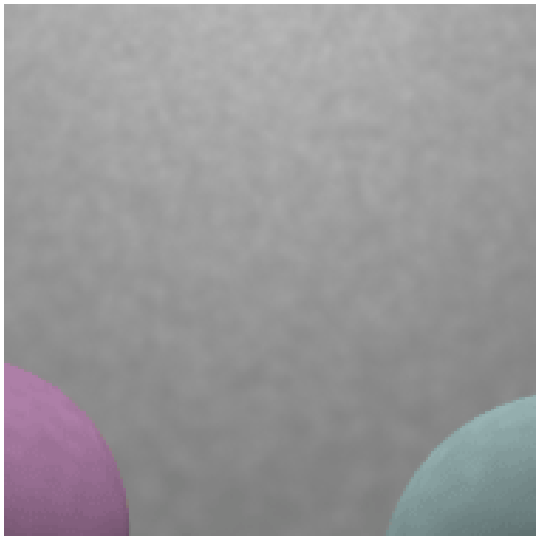
Como puede observarse, el efecto de iluminación directa con NEE tiene un tiempo de ejecución mucho menor y converge mejor (esquinas muy definidas, colores homogéneos, etc.). La rapidez del NEE se debe a que el cálculo que utiliza para estimar la radiancia en un punto es lineal, sin embargo la búsqueda que se realiza en el mapa de fotones, incluso con una estructura relativamente optimizada como lo es el *k-d-tree* utilizado, tiene un coste mucho más elevado.

En *photon mapping* los efectos como cáusticas en materiales especulares y refractantes son muy fáciles de conseguir, pues el camino que realizan los rayos que muestrean la luz parte desde la propia luz puntual. Esto hace que los rayos se comporten de forma más fiel a la realidad, permitiendo que se formen cáusticas realistas (al contrario que en el *path tracing*, cuyas 'cáusticas' en la esfera de cristal se conseguían al reflejar los puntos del techo más iluminados que estaban cerca de la luz).

Sin embargo otros efectos como el de luz directa requieren mucho más tiempo de procesado para conseguir convergencia, ya que se necesitan muchos fotones para conseguir que los materiales difusos tengan un color relativamente homogéneo (como se observa en la comparativa de la figura 3.2) y que las sombras de los objetos (en este caso el de las esferas) estén bien definidas (figura 3.3).

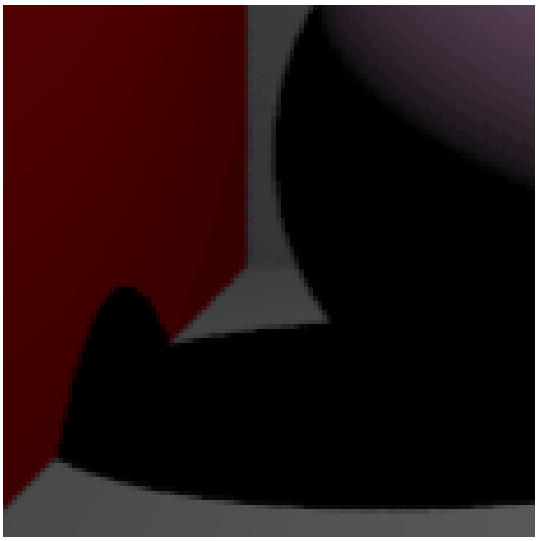
El problema de *photon mapping* para materiales difusos es que podría conseguir antes esa 'homogeneidad' si se aumentara el radio de búsqueda de fotones, sin embargo empeoraría la distinción

(a) Con NEE, zoom en difuso homogéneo

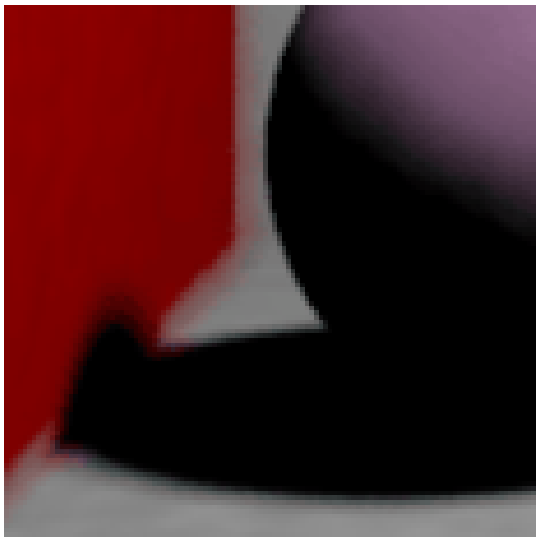


(b) Mapa fotones, zoom en difuso no homogéneo

Figura 3.2: Comparación de materiales difusos



(a) Con NEE, zoom en sombra dura



(b) Mapa fotones, zoom en sombra suave

Figura 3.3: Comparación de luz directa

entre colores distintos (como luz/sombra, intersecciones entre objetos como las esquinas de la habitación, ...).

También cabe destacar que la figura 3.1a con NEE se ve algo más oscura, ya que el cálculo de las radiancias se ve atenuado por el término del coseno. En el mapa de fotones sin embargo, el único cálculo es la multiplicación del flujo y el k_d,

4. BSDFs Delta

Configuramos una escena basada en la caja de Cornell, con dos esferas en su interior: la esfera izquierda con un material de plástico y la esfera derecha con un material de cristal. La iluminación de la escena proviene de una luz puntual situada en la parte superior.

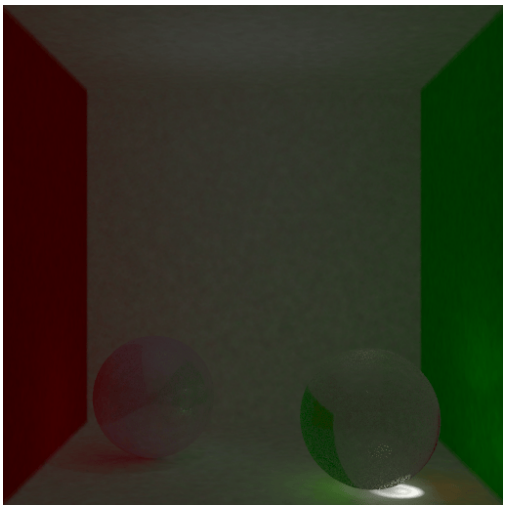


Figura 4.1: Caja de cornell, luz puntual en el techo, sin NEE. Esfera izquierda con k_d = 0,7 y k_s = 0,2. Esfera derecha con k_s = 0,1 y k_t = 0,8. 5M de fotones totales, 16rpp. Fotones globales: 100 vecinos, 0.05 radio. Fotones cáusticos: 0.025 radio

En cuanto a los fenómenos producidos por el *Photon Mapper* en la figura 4.2, notamos que captura con precisión los efectos de cáusticas en el suelo generados por la luz puntual al atravesar la esfera de cristal.

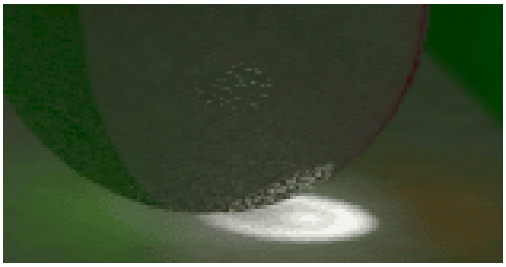


Figura 4.2: Zoom en la cáustica refractante del suelo

También se observa algo de cáusticas en las paredes, cerca de las esferas, provenientes de la especularidad de los materiales (figura 4.3).

Estos fenómenos son computacionalmente eficientes en Photon Mapping porque los fotones que interactúan con las BSDF delta se acumulan en el mapa de fotones, evitando cálculos excesivos en tiempo de renderizado. Además, al emplear el mapa de fotones cáusticos, se pueden configurar con más precisión las cáusticas de la escena.

Figura 4.3: Zoom en la cáustica refractante del suelo

En cuanto a los efectos producidos por el trazado de rayos, destacan las sombras duras (y definidas) que proyectan las esferas en el suelo y la pared adyacente (aunque, debido a la oscuridad de la escena provocada por la intensidad de luz de la cáustica, no se aprecian del todo en la figura 4.1, se aprecian mejor en la figura 7.3a). Estas sombras se generan debido a una combinación de varios factores (figura 4.4): la escasez de fotones en la zona donde se ubica la sombra, el parámetro de radio bajo (de 0.05 en este caso) y el número de fotones lanzados (*número de random walks*).

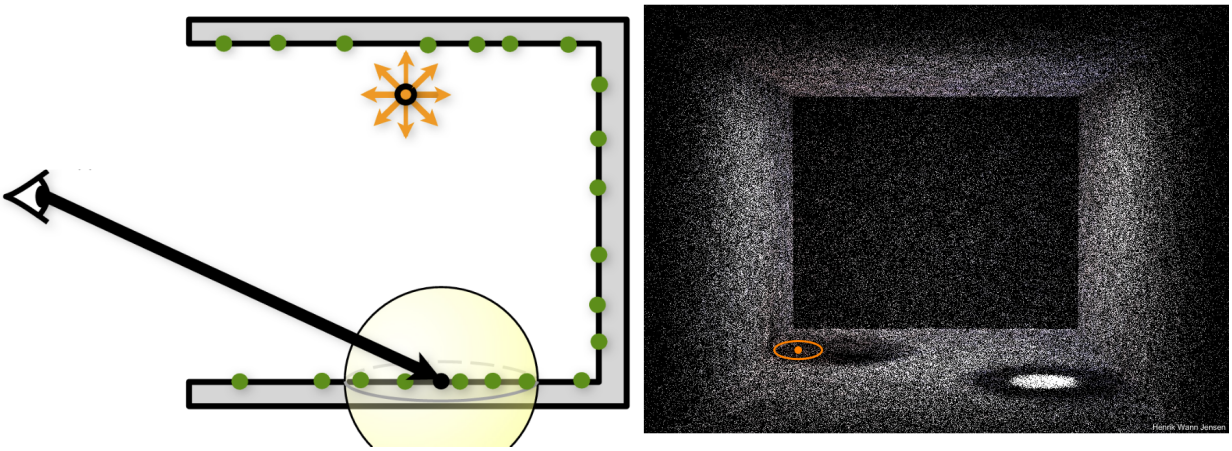
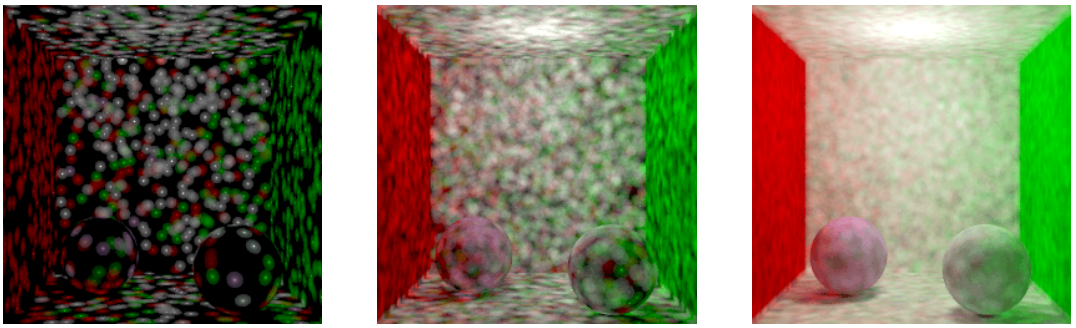


Figura 4.4: Representación de la estimación de una sombra (con centro en el punto naranja y en un radio resultante en la circunferencia naranja

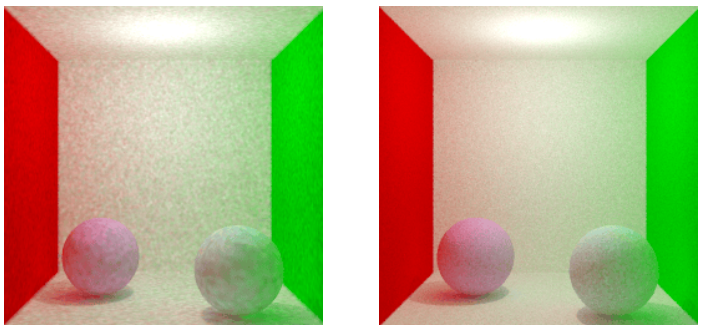
5. Análisis de los parámetros

5.1. Número de fotones

A continuación, en la figura 5.1 se observan unas imágenes renderizadas con un número fijo de vecinos tomados en el paso 2, variando el número de fotones lanzados en el paso 1.



(a) 1k fotones, 2 seg (b) 10k fotones, 5 seg (c) 100k fotones, 23 seg



(d) 1M fotones, 2 min 23 seg (e) 10M fotones, 17 min

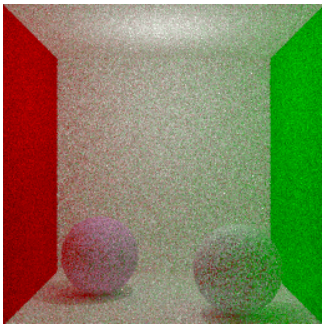
Figura 5.1: Renders con número de fotones creciente, con 16 rpp. Con motivo de mejorar la convergencia, se ha optado por usar k = 100 vecinos y un $Radio_{max} = 0.05$ (para no mezclar mucho los colores).

Como puede observarse en la figura 5.1, la calidad de imagen mejora mucho cuantos más fotones hay y además se reduce el sesgo (la figura 5.1e con 10 millones de fotones es la que mejor se ve). Sin embargo, esto es acosta del tiempo que requiere, pues la búsqueda de fotones del paso 2 tarda cada vez más al haber más fotones almacenados (coste en tiempo polinómico).

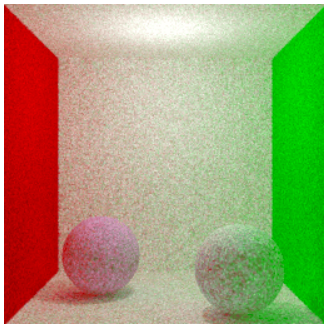
En cuanto a la convergencia, hay varios aspectos que consiguen que la imagen no converja tanto. Por ejemplo: la imagen se ve algo saturada, las sombras de las esfera no son tan oscuras como deberían y el techo está demasiado iluminado por la luz puntual. Al no haber NEE, los colores de la escena se basan solamente en los fotones almacenados, y al no se utilizar el término del coseno, no se atenuan ciertas partes de la escena (como las sombras de las esferas o, en general, los puntos más alejados de la luz).

5.2. Número de vecinos

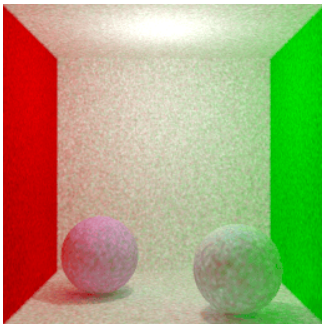
A continuación, en la figura 5.2 se observan unas imágenes renderizadas con un número fijo de fotones lanzados en el paso 1, variando el número de vecinos tomados en el paso 2.



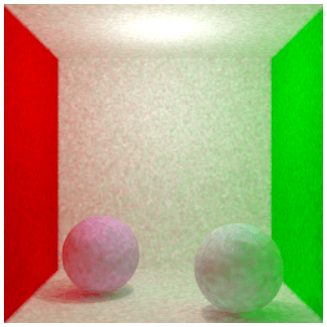
(a) 1 vecino, 1 min 45 seg



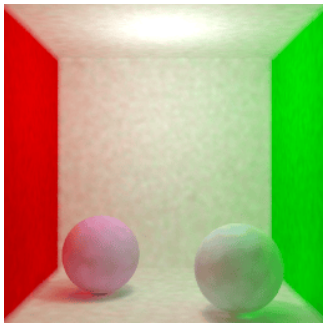
(b) 10 vecinos, 1 min 48 seg



(c) 50 vecinos, 2 min 6 seg



(d) 100 vecinos, 2 min 23 seg



(e) 500 vecinos, 3 min 22 seg

Figura 5.2: Convergencia de materiales con luz de área con NEE, con 16 rpp. Con motivo de mejorar la convergencia, se ha optado por usar k = 1M de fotones y un $Radio_{max} = 0.05$ (para no mezclar mucho los colores).

Como puede observarse en la figura 5.2, la calidad de imagen no mejora con el incremento de vecinos. La mezcla de colores de 500 fotones que se produce en la figura 5.2e provoca que se tomen fotones más alejados del punto que se analiza, provocando que las intersecciones entre planos o las zonas entre la sombra de las esfera y el suelo iluminado queden algo borrosas y menos fieles a cómo deberían ser (se aumenta el sesgo). Sin embargo, la figura 5.2d con 100 vecinos, aunque tenga más imperfecciones en las superficies con color homogéneo (como la pared del fondo), consigue delimitar mejor las esquinas de la habitación y hacer más duras las sombras de las esferas (se reduce el sesgo).

Esto va ligado al coste en tiempo, pues a más vecinos, más tarda en hallarlos en la estructura de almacenaje de los fotones. Aunque el tiempo tampoco varía mucho, pues lo que hace que tarde realmente es el número de fotones almacenados entre los que hay que buscar.

En cuanto a la convergencia, se observa que debería haber un equilibrio en el número de vecinos: Muchos vecinos implica muestrear más fotones lejanos, difuminando demasiado algunas zonas de la escena como las esquinas de la habitación o las sombras de las esferas (como en la figura 5.2e). Pocos vecinos implica muestrear solo los fotones más cercanos, provocando así que (a raíz del color bleeding) salgan puntos de colores en superficies que deberían tener un coloreado más suave (como la pared del fondo en la figura 5.2d).

5.3. Conclusiones

A lo largo de las distintas pruebas realizadas, se observa que la convergencia de la escena no guarda necesariamente una relación directa con el tiempo de ejecución. Aunque un mayor tiempo de renderizado mejora la calidad de la imagen, este aumento es considerablemente mayor en comparación con otros métodos, como el *Path Tracer*, donde es posible obtener un resultado visualmente satisfactorio con un tiempo de ejecución moderado y un nivel de ruido aceptable.

En el caso del *Photon Mapper*, el principal cuello de botella se encuentra en el paso 2 del algoritmo: la renderización de la escena. Durante esta etapa, aunque el proceso principal consiste en calcular las intersecciones de los rayos hasta que chocan con una superficie difusa (o determinar si no chocan con nada) y estimar la ecuación de renderizado utilizando la densidad de fotones, el mayor coste computacional proviene de la búsqueda de estos fotones en la estructura *k-d tree*. Esta operación, que implica localizar los fotones más cercanos en función de los parámetros configurados, resulta ser la principal limitación en términos de tiempo de ejecución en los casos donde hay muchos fotones almacenados (muchos fotones entre los que buscar).

En resumen, aunque el *Photon Mapping* ofrece una capacidad notable para representar efectos complejos de iluminación global, como cáusticas y *color bleeding*, su eficiencia está condicionada por la gestión y consulta de los fotones en la estructura de datos, lo que plantea un desafío para escenas con un alto número de fotones o una mayor resolución.

6. Diseño de la ruleta rusa

En la ruleta rusa original, siempre se tiene en cuenta un cierto coeficiente de absorción, ya que todos los materiales tienen un cierto porcentaje de absorber el rayo (en este caso, un 10 %). Esto es especialmente útil a la hora de generar fotones en el del paso 1 de *photon mapping*, ya que permite simular cómo se comportan los materiales en realidad, y además añade una condición terminal a los *random paths*, pues sino cada camino seguiría rebotando hasta el infinito.

Sin embargo en el paso 2, es interesante no tener absorción, pues se quiere llegar al evento difuso para obtener los fotones vecinos, y el evento absorbente no aporta mucho a esta estimación. Por ello, se han querido estudiar varios escenarios:

1. Se muestrea solo 1 vez el evento, si toca absorbente se termina el camino y se pinta de negro. Consecuencias:

- Si 1 rpp: habrá píxeles completamente negros donde no debería. Figura 6.1a.
- Si >1 rpp: todos los píxeles que no sean luz de área podrían verse más oscuros en función de la suerte de haber tocado un evento absorbente. Figura 6.1c.

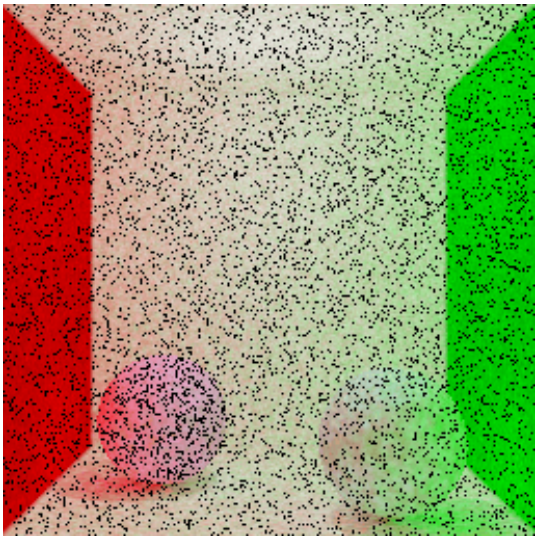
2. Se muestrean eventos iterativamente hasta que no ocurre ningún evento absorbente. Consecuencias:

- Si 1 rpp: ya no hay píxeles negros donde no debería. Figura 6.1b.
- Si >1 rpp: los píxeles no se ven más oscuros. Figura 6.1d.
- Se complica innecesariamente el código.
- Aumenta muy sutilmente el tiempo de ejecución (porque la probabilidad de absorbente es siempre 10 %) a cambio de una mejor convergencia.

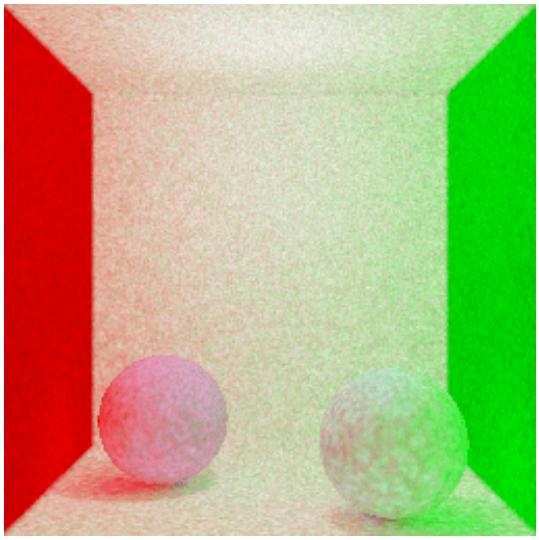
3. Igual que el caso anterior, pero alterando las probabilidades de la ruleta rusa. Se provoca que no sea posible un evento absorbente. Consecuencias:

- Si 1 rpp: ya no se ven píxeles negros donde no debería. Figura 6.1b.
- Si >1 rpp: los píxeles no se ven más oscuros. Figura 6.1d.
- Muy limpio y fácil de implementar.
- Tiempo de ejecución no aumenta porque el coste es de prácticamente 1 ciclo el 10 % de las veces. El motivo es que solo hay que cambiar una suma que corresponde al denominador de la probabilidad de muestrear un tipo de evento, en la función de la ruleta rusa.

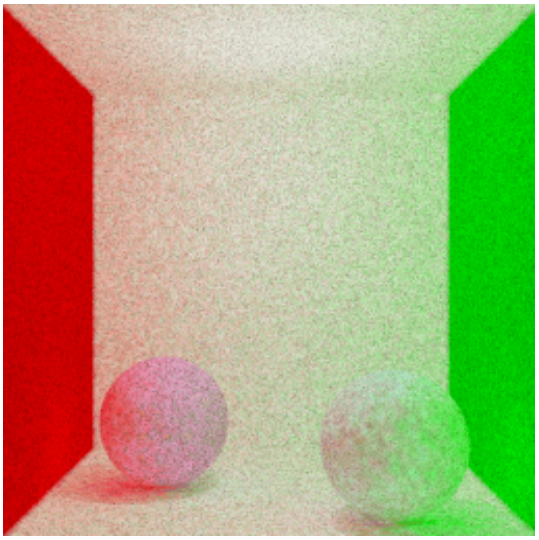
Finalmente, **se ha optado por la tercera opción**, reflejada en el código entregado.



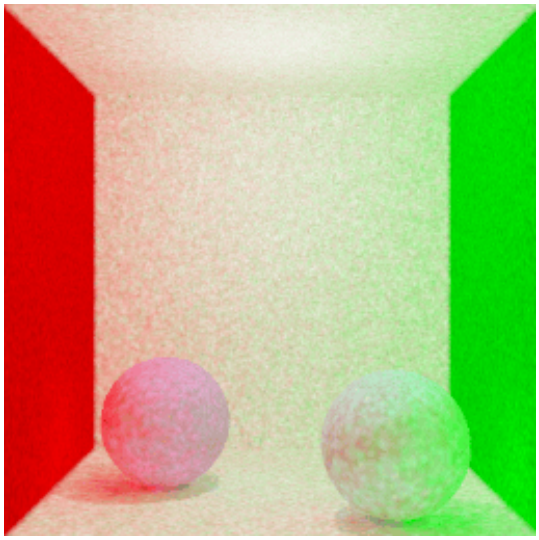
(a) 1 rpp, con absorción



(b) 1 rpp, sin absorción



(c) 16 rpp, con absorción



(d) 16 rpp, sin absorción

Figura 6.1: Comparación de escenas con y sin absorción, con distinto rpp

7. Extensiones

7.1. Kernels adicionales

Para la estimación de densidades, se han implementado varios kernels avanzados[3]. La figura 7.1 ilustra cómo cada kernel pondera la contribución del flujo de los fotones en función de su distancia normalizada al punto de intersección del rayo ($u = r/r_{max}$), permitiendo analizar y comparar su comportamiento.

La figura 7.2, junto con la gráfica 7.1, permite observar que los kernels Gaussiano, Epanechnikov,

[3]Más información en https://en.wikipedia.org/wiki/Kernel_(statistics)#

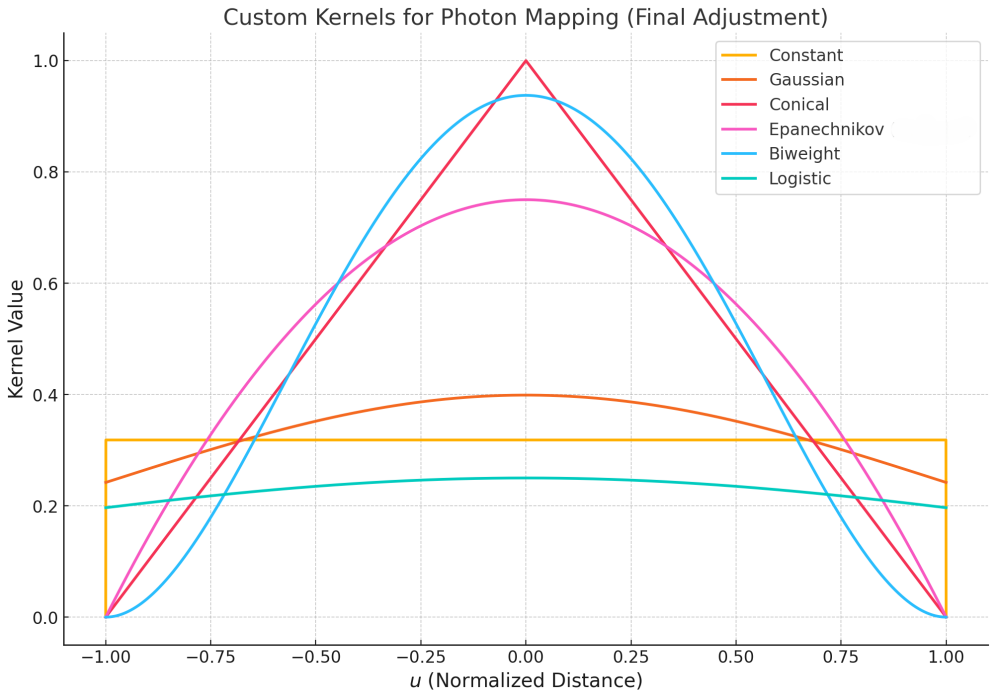


Figura 7.1: Valores del kernels vs distancia al centro normalizada u (r / r_{max}), $r_{max} = 0{,}05$

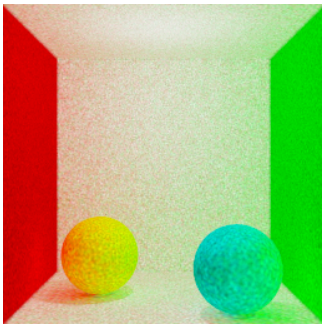
Logístico e incluso Constante generan un efecto suavizantes. En cambio, los kernels Bipeso y Cónico tienden a producir imágenes menos homogéneas.

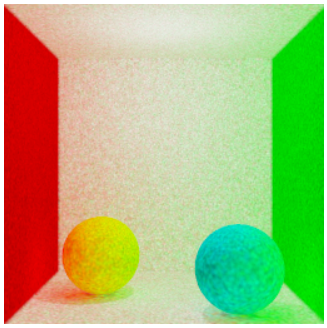
Aunque el kernel Epanechnikov es teóricamente óptimo para la estimación de densidades[4], en la práctica observamos que el kernel Gaussiano produce resultados generalmente superiores. Esto se debe a que el efecto de su ponderación más suave y gradual da lugar a un aspecto más realista y difuso.

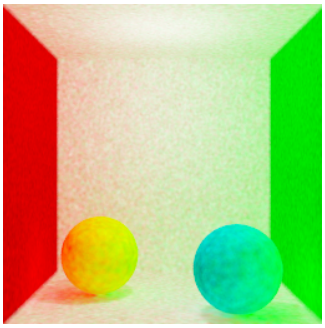
Por otro lado, los kernels Bipeso y Cónico muestran un rendimiento ligeramente superior en detalles como las esquinas de las paredes, donde logran preservar mejor los contrastes y capturar mejor la transición de una pared a otra. Esto se debe a que estos kernels proporcionan mayor discriminación en la distancia, asignando un peso más significativo a los fotones más cercanos.

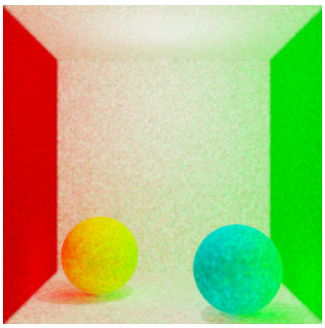
La inspiración ha sido obtenida gracias al repositorio de referencia, número [4] en la bibliografía.

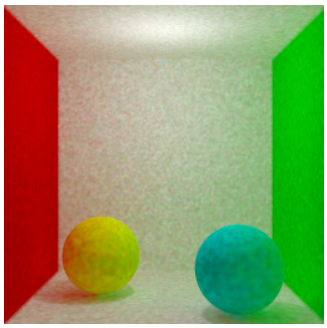
[4]Fuente: Kernel Density Estimation, https://phas.ubc.ca/~oser/p509/Lec_23.pdf


(a) Bipeso


(b) Cónico


(c) Constante


(d) Epanechnikov


(e) Gaussiano

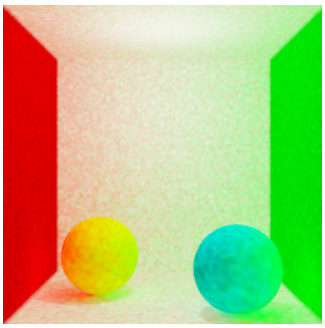
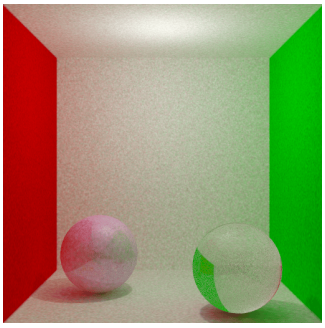

(f) Logístico

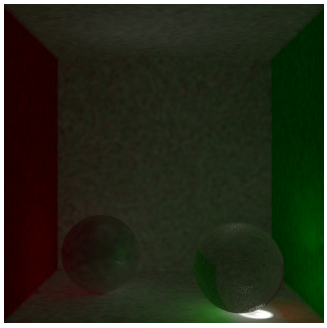
Figura 7.2: Comparación de kernels con caja de cornell todo difuso

7.2. Mapa de fotones cáusticos

Se ha separado el mapa de fotones en fotones globales (aquellos que solo han pasado por superficies difusas) y cáusticos (los que han pasado por una superficie especular o refractante) para poder modificar el muestreo de los fotones en cada caso.

Se ha realizado una prueba con 100 vecinos y 0.05 de radio para los fotones globales y un radio de 0.025 para los cáusticos, que se muestra en la figura 7.3.


(a) Solo fotones globales


(b) Solo fotones cáusticos

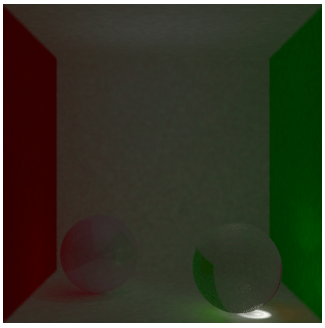

(c) Fotones globales + cáusticos

Figura 7.3: Comparación de mapas de fotones globales y cáusticos

En la figura 7.3a vemos que la escena realmente se ve bastante bien (la escena se ve con más claridad y las sombras de las esferas se distinguen más que en la figura 7.3c, por ejemplo). Se podría

obtener entonces un resultado total mejor (sin que los cáusticos tomen valores tan altos) poniendo un límite al número de fotones cáusticos que se toman.

La inspiración también ha sido obtenida gracias al repositorio de referencia mencionado en la bibliografía ([4]).

8. Carga de trabajo y metodología

Se utilizó un repositorio en GitHub para gestionar el código de manera compartida y registrar el progreso del proyecto. Los commits se emplearon como un mecanismo para documentar las tareas, ya fueran pendientes o completadas. Aunque el trabajo fue mayoritariamente individual, con tareas asignadas a cada miembro, también se realizaron sesiones conjuntas en momentos clave, especialmente cuando era necesario modificar código crítico o abordar decisiones importantes.

Además, se elaboró una lista de prioridades que seguimos estrictamente para las extensiones implementadas. Esto nos permitió mantener el enfoque en los aspectos más relevantes del proyecto y evitar desviarnos hacia objetivos secundarios.

Módulo	**Manel**	**Ming Tao**
Photon Tracing	10h	1,5h
Estimación de densidades	7h	3h
Kernels adicionales	2h	2h
Mapa de fotones cáusticos	2h	0h
Paralelización	0h	1h
Adaptación módulos de Path Tracer	2h	0h
Memoria	7h	8h
Total	30h	15,5h

Tabla 8.1: Análisis de carga de trabajo

9. Bibliografía

[1] *Kernel Density Estimation*. URL: https://phas.ubc.ca/~oser/p509/Lec_23.pdf. (accessed: 10.01.2025).

[2] *Kernels (statistics)*. URL: https://en.wikipedia.org/wiki/Kernel_(statistics)#. (accessed: 10.01.2025).

[3] *Repositorio de referencia de José Daniel Subías*. URL: https://github.com/dsubias/IG-mini-PM. (accessed: 10.01.2025).

[4] *Repositorio de referencia de Julia Guerrero Viu*. URL: https://github.com/juliagviu/ComputerGraphics. (accessed: 10.01.2025).

[5] *Transparencias de teoría de Informática Gráfica*. URL: https://moodle.unizar.es/add/pluginfile.php/12027972/mod_folder/content/0/11-photon_mapping.pdf. (accessed: 09.01.2025).

[6] *Transparencias y guión de las prácticas de laboratorio de Informática Gráfica*. URL: https://moodle.unizar.es/add/pluginfile.php/12027997/mod_label/intro/05-photonmapping_es.pdf?time=1731494699308. (accessed: 09.01.2025).